MŰEGYETEM 1782

SZAKDOLGOZAT FELADAT

Somlai Zoltán
mérnökjelölt részére

Frekvenciamérés és -követés nemlineáris Kalman-szűrővel

A jelfeldolgozás egyik klasszikus feladata a mért jel frekvenciájának meghatározása. Az utóbbi évtizedekben különféle alkalmazási területeken számos algoritmus született a probléma megoldására. Az akusztikai, zenei alkalmazásokban gyakori eset, hogy a jel amplitúdója és frekvenciája a periódusidőhöz képest lassan változik. Ekkor olyan frekvenciamérő eljárás alkalmazása szükséges, mely a lassú változásokat is képes követni. A lineáris dinamikus rendszerek esetén optimális Kalman-szűrő lehetőséget kínál erre, ugyanakkor ebben az esetben kihívást jelent a probléma nemlineáris természete.

Jelen szakdolgozattéma a Kalman-szűrővel megvalósítható frekvenciamérő és -követő eljárások vizsgálatára fókuszál. Ezeknek a módszereknek az előnye, hogy a becslés vélt pontosságáról is információt kapunk. Emellett a felhasznált jelmodell alapján a frekvencia meghatározásán túl a teljes becsült jelet is előállíthatjuk, ami az így keletkező hibajel további feldolgozására is lehetőséget adhat.

A hallgató feladatának a következőkre kell kiterjednie:

- A szakirodalom feldolgozásával ismerje meg és ismertesse a frekvenciamérésre alkalmas nemlineáris Kalman-szűrő működési elvét.
- Készítsen saját Kalman-szűrő implementációt, mutassa be a működését és a változtatható paramétereket.
- Mesterségesen előállított tesztjelek segítségével igazolja, hogy a megvalósított szűrő a várakozásoknak megfelelően működik.
- Vizsgálja meg a frekvenciakövető eljárás működését valódi zenei jeleken is. Mutassa be a szűrő paramétereinek hatásait a kapott eredményekre. Az eredményeket hasonlítsa össze legalább egy másik kiválasztott frekvenciamérő módszerrel.
- Bővítse az eredeti algoritmust úgy, hogy az képes legyen egy jel több komponensének egyidejű mérésére, követésére.

Tanszéki konzulens: Dr. Rucz Péter, tudományos munkatárs

Budapest, 2020.09.15.

Dr. Imre Sándor

egyetemi tanár
tanszékvezető

Konzulensi vélemények:

Tanszéki konzulens: □ Beadható, □ Nem beadható, dátum: aláírás:

Külső konzulens: aláírás:

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Hálózati Rendszerek és Szolgáltatások Tanszék

1117. Budapest, Magyar tudósok körútja 2. I. Ép. B. 121.
Telefon: 463-3261 Fax: 463-3263
URL: http://www.hit.bme.hu

Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Hálózati Rendszerek és Szolgáltatások Tanszék

Somlai Zoltán

Frekvenciamérés és -követés nemlineáris Kalman-szűrővel

KONZULENS

Dr. Rucz Péter

BUDAPEST, 2020

Tartalom

HALLGATÓI NYILATKOZAT

Alulírott **Somlai Zoltán**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2020. 12. 10.

.....
Somlai Zoltán

Összefoglaló

A frekvencia meghatározása alapvető fontosságú a jelfeldolgozásban. Periodikus jeleket az amplitúdójuk és a fázisuk mellett a frekvenciájuk különböztet meg. A zenei hangokhoz társított hangmagasságot is a frekvencia alapján határozzuk meg. A változatos alkalmazásoknak köszönhetően sokféle algoritmus született különféle jelek frekvenciájának meghatározására, követésére. Jelen dolgozatban egy Kalman-szűrő megvalósítása, optimalizálása, fejlesztése a feladat.

Ahogy sok mérnöki probléma esetében, úgy a frekvenciamérésnél is nemlineáris rendszerek segítségével modellezhetjük a valóságot. Az eredeti Kalman-szűrővel szemben, amely csak lineáris rendszermodellek leírására optimális, a kiterjesztett ECKF (Extended Complex Kalman Filter) a munkaponti linearizálás módszerével a nemlineáris modell kezelésére is alkalmas, megtartva az eredeti szűrő előnyös tulajdonságait. Nagy előnye a Kalman-szűrőn alapuló módszereknek, hogy a méréseket sorozatosan átlagolja a modellszámításokkal, így a becslésből származtatott kimenő adatok képesek követni a megfigyelt rendszer dinamikáját.

A dolgozatban részletesen tárgyalásra kerül az ECKF működése, az, hogy milyen feltételezésekkel élünk, a szűrő implementálása Matlab programnyelven és a tesztelés. A megvalósított algoritmus lépéseit külön-külön tárgyalom. Megvizsgálom egyszerű mérőjelek frekvenciájának követését különböző zajszintek mellett, illetve különböző zenei hangok feldolgozását is. Az algoritmust egy másik népszerű frekvenciamérő módszerrel is összehasonlítom. A szűrő fejlesztéseképp az alapjel frekvenciájának követése mellett megvalósítom egy azzal egyidejű keskenysávú zajkomponenst követését is. Ez a lehetőség, az alkalmazástól függően további információt szolgáltat, a visszaállítását még pontosabbá teszi.

Abstract

Frequency measurement is an essential task in the field of signal processing. Besides the amplitude and the phase periodic signals are distinguished by their frequency. The pitch associated to musical sounds are determined by the frequency as well. Thanks to the variety of applications, many kinds of algorithms were devised in order to measure and track frequency. The objective of this thesis is to implement, optimize and extend a Kalman-filter-based frequency tracker by further enhancements.

As in the majority of engineering problems, in frequency measurement the modelling of reality is only possible by means of nonlinear systems. Unlike the original Kalman filter which is only optimized for linear system models, the extension called ECKF (Extended Complex Kalman Filter) is capable of handling nonlinear system models with the method of work point linearization while simultaniously keeping the advantegous properties of the original filter. An advantage of Kalman-Filter-based methods is that they average the meauserements with the model calculations iteratively, thus, the estimated state is able to track the dynamics of the observed system.

In this thesis the operation of the ECKF will be discussed in detail as well as the presumptions used, the implementation of the filter in the Matlab programming language and testing. The steps of the implemented algorithm will be explained separately. I will examine how the filter tracks simple measuring signals as well as musical tones. I will compare the algorithm with an other popular frequency measuring method. As a development of the filter, I will implement a way to track a narrowband noise component simultaneously with the fundamental component. This provides an opportunity to obtain more information as well as making the reconstruction of the observed signal more precise depending on the application.

1 Bevezető

A frekvenciamérés általános jelfeldolgozási feladat, széles körben kutatott. Számos algoritmus született a feladat megoldására. Az egyszerűbb hangok frekvenciájának meghatározásán felül szerepet kap az algoritmus zenei hangok esetében is. Hangmagasság detektálása a zenében nagy jelentőséggel bír. A hangmagasságot a zenei hang frekvenciája határozza meg, a nagyobb frekvenciát halljuk magasabbnak. További felhasználási lehetősége a frekvenciakövetésnek a beszédfelismerésben van, de ezt nem tárgyalom a dolgozatomban.

A dolgozat célja egy működő Kalman-szűrő implementáció létrehozása Matlabban, mely képes egy jelnek az alapfrekvenciáját az adott pillanatban meghatározni. Ebben a dolgozatban a feldolgozott jelek általában audiojelek. Fontos szempont az, hogy amikor új hang jelenik meg (az alapfrekvencia megváltozik) akkor minél gyorsabban (valós időt megközelítve) detektálja ezt a szűrő, minél kisebb kilengésekkel. Azonban ezt nem mindig egyszerű kivitelezni, több nehezítő tényező is felléphet. A zenében az alapfrekvencia gyakran a periódusidő többszöröséig megegyezik, majd hirtelen vált. A zenei hangok gyakran nem tekinthetőek harmonikusnak, mely a hangmagasság detektálását nehezíti. Ha vannak is felharmonikusok, a különböző hangszerek esetében más mennyiségben vannak jelen. A valódi zenei hangok amplitúdója, frekvenciája változik időben akár ugyanazon hang megszólaltatásánál is, mert az ember nem tudja minden pillanatban ugyanúgy játszani (pl. a furulya nyílásába a levegőt kicsit máshogy fújja). Mivel a hangok nem ideálisak, ezért zajjal vannak terhelve, ami szintén megnehezíti a feladatot.

Az ECKF szűrő többféle előnyös tulajdonsággal rendelkezik más frekvenciakövető algoritmusokkal szemben. Mintáról mintára ad becslést, mely, szemben más algoritmusokkal, optimális ahhoz, hogy valós időben követni tudja a frekvenciát. Késleltetése alacsony, zajtűrése nagy. A felhasznált környezet a Matlab, mely optimális algoritmusunk implementálására és eredményeinek ábrázolására és viszonylag egyszerű használatot biztosít.

A dolgozatom felépítését röviden összefoglalom. A második fejezetben az ECKF-t írom le. Kitérek arra, hogy milyen modellt használunk a bejövő hangra és hogy milyen feltételezésekkel élünk ezzel kapcsolatban. Részletezem a Pitch detector működést, amely meghatározott időkeretenként detektálja a hangban a változást (új hang szólal meg). Az ötödik fejezetben a Matlab-implementációt írom le, az algoritmus lépéseire külön-külön kitérve. Szó

lesz benne az ECKF több komponens követésére való fejlesztéséről is. A hatodik fejezetben a tesztelés módját, tapasztalatait írom le a levont következtetésekkel együtt. Végül egy rövid összefoglalóval zárom a dolgozatot.

2 ECKF

Ebben a fejezetemben nagyrészt az alábbi cikkekre támaszkodom forrásként: [1] [2]

2.1 Mi is az ECKF?

Extended Complex Kalman Filter a betűszó feloldása, melyből látszik, hogy egy Kalman-szűrőről van szó. A Kalman-szűrő egy olyan algoritmus, mely alapvetően két lépésből áll: becslés és korrigálás. A szűrő először mindig egy becslést ad az állapotváltozókra, illetve az azokat terhelő bizonytalanságokra egy előre definiált modell alapján, ezt hívjuk becsült állapotnak. Ezután mérés alapján frissíti a változókat, súlyozza a becsült és a mért értékeket. Ez a súlyozás a kovarianciától függ. A kovariancia két változó együtt mozgását jelenti, a Kalman-szűrőnél ez az állapotok becsléséből és a bizonytalanságukból származik. Az új állapotbecslés értéke mindig a becsült és a mért állapot között van. A korrigálás során az állapotváltozót és a hiba kovarianciát is frissítjük. Ez egy iteratív jellegű algoritmus, mindig ugyanazok az iterációk ismétlődnek benne, csak frissített, pontosabb eredményekkel. A számításoknál mindig a közvetlenül előző lépés eredményeit használja az algoritmus, a legfrissebbet, legpontosabbat, nincs szükség az összes lépés eredményeinek tárolására. A sorozatos mérések és frissítések miatt a szűrő jobb eredményt ad, mintha minden időpontban szimplán mérnénk az állapotváltozót. [3]

Alapvetően a Kalman-szűrőt lineáris dinamikus rendszerek állapotainak becslésére találták ki, de a mi rendszerünk nemlineáris, mert az állapotváltozók és a mérendő mennyiségek között nemlineáris kapcsolat áll fenn. Ez azt jelenti, hogy a jel mintából előállítani az amplitúdót, frekvenciát és a fázist egy nemlineáris modellel lehet csak. A betűszóban található „Extended” (kiterjesztett) kifejezés ennek a problémának a megoldására utal, ugyanis kiterjesztjük a Kalman-szűrőt nemlineáris dinamikus rendszerek követésére. Tulajdonképpen így a Kalman-szűrő nemlineáris változatát kapjuk, melyben linearizálva van az a modell, ami alapján becslünk. A kiterjesztett Kalman-szűrő algoritmus bonyolultsága, számításigénye kisebb, mint más nemlineáris szűrők, széles körben elterjedt a használata.

Az alfejezet utolsó bekezdésében a betűszóban található „Complex” (komplex) kifejezés mivoltát fejtem ki. Sokféle modell van használatban egy jel frekvenciájának, amplitúdójának, fázisának követésére, mi itt egy komplex modellre támaszkodunk. Ez azt jelenti, hogy a számítások a komplex számtartományra vannak kiterjesztve, az állapotvektor is komplex. A

komplex modell használatának nagy előnyei vannak az eltorzult jelek követésében. A szinuszos jeleket általában komplex amplitúdóval jellemzünk, és az ECKF-ben használt modellünk is szinuszos jelen alapszik. [4]

2.2 Milyen feltételezésekkel élünk?

A vizsgálandó jel modellje egy szinuszos hasznos jelből és a hozzáadott zajból áll. A szinuszos jellel való modellezés egy általános periodikus jel modelljének ésszerű egyszerűsítéséből származik. Vegyük először az általános modellt:

$$y_k = \sum_{i=0}^{N} a_i \cos(\omega_i t_k + \Phi_i) + v_k \tag{2.1}$$

A bemeneti jelet y_k-val jelöljük, k jelöli a diszkrét időpillanatot. Az amplitúdót a_i-val a fázist Φ_i-vel jelöljük. Az ω_i körfrekvenciát 2π-vel leosztva f_i frekvenciát kapjuk. A zaj modellje (mérési zaj) normál eloszlású Gaussi fehérzaj 0 várható értékkel és σ_v szórással jelölése: v_k. A normál eloszlást a CHT (centrális határeloszlás tétel) alapján feltételezhetjük, mert a zaj több eltérő frekvenciájú és intenzitású jel zavaró összessége és sok független változó középértéke jó közelítéssel normál eloszlású. Mivel a feladat célkitűzése az volt, hogy az alapfrekvenciát meghatározzuk, és a harmonikus összetevők csak kis hányadát adják a frekvenciakomponenseknek, így egyszerűsíthetjük a (2.1) egyenletet:

$$y_k = a_1 \cos(\omega_1 k T_s + \phi_1) + v_k \tag{2.2}$$

Itt már csak az alapjel amplitúdója, frekvenciája és fázisa jelenik meg, T_s a mintavételi periódusidő.

Algoritmusunk követi az alapfrekvenciáját a jelnek, de az amplitúdóját és fázisát is figyeli. Éppen ezért az állapotvektort így definiálhatjuk:

$$x_{k=} \begin{bmatrix} \alpha \\ u_k \\ u_k^* \end{bmatrix} \tag{2.3}$$

amiben:

$$\alpha = \exp(j\omega_1 T_s)$$

$$u_k = a_1 \exp(j\omega_1 k T_s + j\Phi_1)$$

$$u_k^* = a_1 \exp(-j\omega_1 k T_s - j\Phi_1) \tag{2.4}$$

Az állapotvektor frissítése egy nemlineáris függvény segítségével történik:

$$\begin{bmatrix} \alpha \\ u_{k+1} \\ u^*_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & \frac{1}{\alpha} \end{bmatrix} \begin{bmatrix} \alpha \\ u_k \\ u^*_k \end{bmatrix}$$

$$x_{k+1} = f(x_k) + w_k$$

$$f(x_k) = \begin{bmatrix} \alpha & \alpha u_k & \frac{u^*_k}{\alpha} \end{bmatrix}^T \tag{2.5}$$

Ahol w_k-val jelöljük a folyamat zaját, amelyet Gaussi fehérzajnak feltételezünk.

A bejövő jel és az állapotvektor kapcsolatát a következő egyenletek szerint írjuk le:

$$y_k = Hx_k + v_k$$

$$H = [0 \quad 0.5 \quad 0.5] \tag{2.6}$$

H a megfigyelési mátrix.

Mivel nemlineáris a probléma, ezért kiterjesztett Kalman-szűrőt alkalmazunk, melyben a linearizálás úgy van megoldva, hogy a nemlineáris függvény Jacobi-mátrixát vesszük, amely annak elsőrendű parciális deriváltjait tartalmazó mátrix:

$$F_k = \left.\frac{\partial \mathrm{f}(x_k)}{\partial x_k}\right|_{x_k=\hat{x}^+} = \begin{bmatrix} 1 & 0 & 0 \\ \hat{x}^+(2) & \hat{x}^+(1) & 0 \\ -\frac{\hat{x}^+(3)}{(\hat{x}^+(1))^2} & 0 & \frac{1}{\hat{x}^+(1)} \end{bmatrix} \tag{2.7}$$

Az ECKF egyenletei a következőek:

$$K_k = \hat{P}^- H^{*T} \left[H \hat{P}^- H^{*T} + \sigma_v^2 \right]^{-1} \tag{2.8}$$

A (2.8) a Kalman-nyereség kiszámítására vonatkozó egyenlet, amelynek abban a súlyozásban van szerepe az állapotfrissítésnél, hogy a modellre vagy mérésre támaszkodunk-e jobban. $\hat{P}$ jelöli a becsült állapotvektor kovariancia mátrixát. Ez azt jelenti, hogy a szűrő megbecsli az állapotvektor bizonytalanságát.

$$\hat{x}^+ = \hat{x}^- + K_k(y_k - H\hat{x}^-) \tag{2.9}$$

A (2.9) az állapotváltozót frissíti a mérés alapján, ahol az előbb említett Kalman-nyereség alapján súlyozza az eredményt. ($y - Hx$ a hibajel)

$$\hat{x}^- = f(\hat{x}^+) \tag{2.10}$$

A (2.10) a modell alapján frissíti az állapotváltozók becslését a nemlineáris függvény segítségével.

$$\hat{P}^+ = (I - K_k H)\hat{P}^- \tag{2.11}$$

A (2.11) a kovariancia mátrix becslését frissíti a mérés alapján, hasonlóan a (2.9)-hez.

$$\hat{P}^- = F_k \hat{P}^+ F_k^{*T} + \sigma_w^2 I \tag{2.12}$$

A (2.12) a modell alapján frissíti a kovariancia mátrix becslését, a nemlineáris függvény Jacobi-mátrixa (F) segítségével, a (2.10)-hez hasonlóan. I egy 3x3as egységmátrix.

A leírásban a felsőindexbe rakott + és – jel azt jelenti, hogy a változó a posteriori, frissített állapotáról van szó(+) vagy az a priori, becsült állapotáról(–). Az a priori eset azt jelenti, hogy a bemeneti (y) jel mérése egy lépéssel előbb történt, az előző iteráció eredményét használjuk fel a jelenlegi becslésre. Az a posteriori esetben a bemeneti jel már le lett mérve, frissítve lett az adott időpillanatban, ezekkel az információkkal adtunk becslést. A „kalap” operátor pedig azt jelenti, hogy egy változónak a becsült értékéről van szó: $\hat{x}$ az x becslése. A mérési zaj szórását σ_v-val jelöljük, jelen számításokban értéke 1. A folyamat zajának szórását σ_w-val jelöljük, ami egy feltételezett zaj, amit attól függően állítunk be, hogy mennyire tartjuk helyesnek a modellt. σ_v és σ_w aránya határozza meg azt, hogy a mérést vagy a modellt vesszük nagyobb súllyal az átlagolásnál.

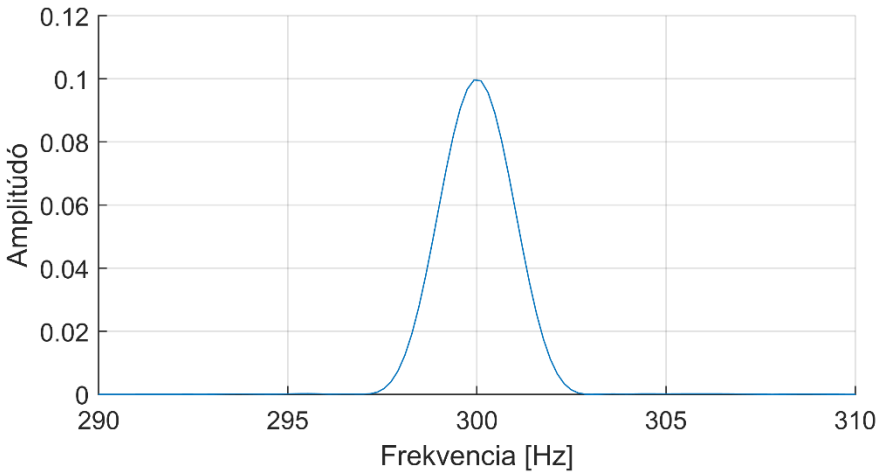
2.3 A pitch detector elve

Algoritmusunkban fontos szerepe van annak, hogy detektálni tudjuk, hogy mikor van alapfrekvencia váltás, azaz mikor jelenik meg új hang. Külön vizsgálandó az az eset, amikor semmilyen hang nem szólal meg, maximum a zaj detektálható, ezek lesznek a csendes részek. Az ilyen csendes részek miatt a jelet kb. 40ms-os, egymást át nem fedő keretekre osztjuk fel, minden keretben egy előre megválasztható számú mintával (a tesztek során általában 2048, ez

a blokkméret). A keret időhossza a mintavételi frekvenciától és a blokkmérettől függ. A pitch detector ilyen keretenként határozza meg az alapfrekvenciát.

A pitch detector egy adott kereten keresi az alapfrekvenciát, ehhez a keret spektrumának vizsgálatára van szükség. A spektrumkép kialakításához a gyors Fourier transzformációt (FFT-Fast Fourier Transform) alkalmazzuk, amelyben az FFT pontok sűrűségét a mintavételi frekvencia és a keret hossza határozza meg. A jel DC összetevőjét eltávolítjuk, mert nem szolgáltat információt az alapfrekvencia szempontjából. A spektrumszivárgás csökkentése érdekében a jelet összeszorozzuk egy, a kerethez méretezett Blackman-ablakkal. Mivel szimmetrikus a spektrum a valódi jeleknél, ezért a negatív frekvenciáknál lévő értékek redundánsak, egyoldalúvá tesszük a spektrumot (csak a pozitív frekvenciaértékeknél vizsgáljuk a csúcsokat).

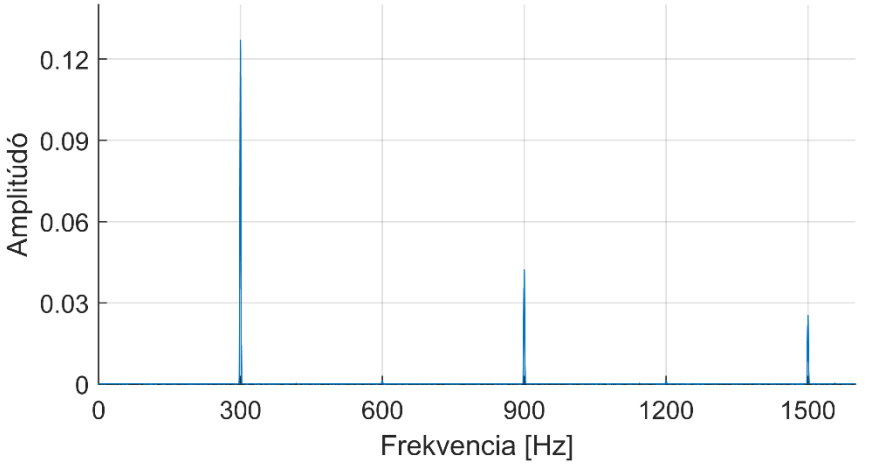
Egy kereten belül az alapfrekvencia rendszerint a legnagyobb amplitúdóval rendelkező csúcs a spektrumábrán, de ez nem minden esetben igaz, ezért más metódust alkalmazunk. A frekvenciaspektrumban egy csúcs magassága jelenti az amplitúdót (ez egy meglehetősen pontos megfeleltetés, mert a spektrum abszolút értékét vesszük) és a csúcs pozíciója jelenti a frekvenciát. Megkeressük először az összes csúcsot. Mivel az algoritmusunk egy bizonyos megadott frekvenciahatárokon belül keresi az alapfrekvenciát, így a határokon kívül eső csúcsokat eltávolítjuk. Ezután csökkenő sorrendbe tesszük a csúcsokat, és a felhasználó által megadható num_peaks paraméternek megfelelő számú legnagyobb csúcsot tartjuk meg. Még az a kikötés van ezekre a csúcsokra, hogy a legnagyobb csúcsnál egy adott számmal (pl. 40dB) kisebb nagyságrendű amplitúdójú csúcs kiesik, ezután vizsgáljuk a megmaradó csúcsokat. Kétféle eset lehetséges: egy vagy több csúcs marad. Ha egyetlen csúcs marad, akkor annak a csúcsnak a frekvenciája az alapfrekvencia. A 2.3 ábra ezt az esetet szemlélteti.



2.3. ábra: Szinuszjel spektruma

Az ablakozás és a nemkoherens mintavételezés miatt a kapott spektrumkép nem tökéletesen vonalas, de jó megközelítéssel a legmagasabb pont 300 Hertznél van. Az algoritmus ezt az egy csúcsot tartja meg, a végeredményhez is ezt az értéket használja fel.

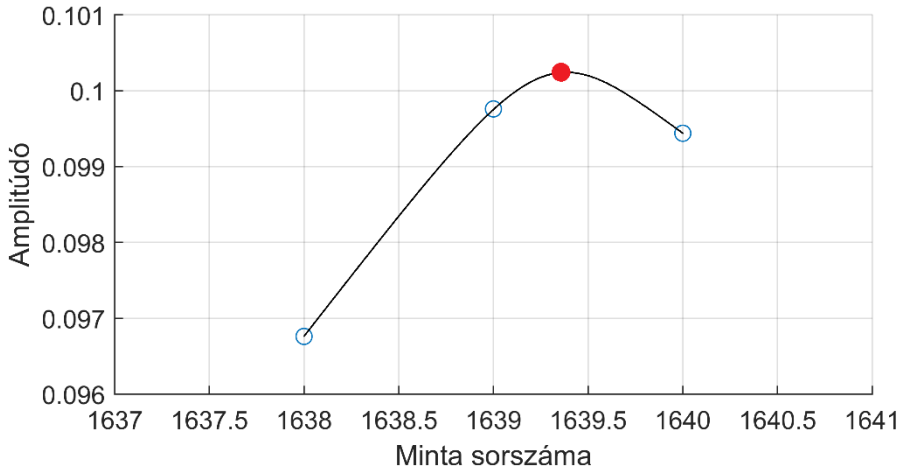
Több csúcs esetén a megjelenő harmonikusok közötti távolság alapján határozzuk meg az alapfrekvenciát. A következő speciális esetben az alapharmonikus frekvenciájának a kétszerese a különbség a csúcsok között, ezt a 2.4. ábra mutatja.



2.4. ábra: Négyszögjel spektruma

Leolvasható, hogy 300, 900 és 1500 Hertznél vannak a csúcsok, különbségük 600 Hertz, ami valóban kétszerese az alapfrekvenciának.

Mivel az FFT-nél a pontosságnak korlátot szab a felbontás, finomítanunk kell a kapott alapfrekvencia eredményt. A megtalált csúcs mellett vesszük a jobb és bal oldalt lévő legközelebbi mintákat és parabolát illesztünk a 3 pontra. Ez a parabola mindig egyértelműen meghatározható, az alapfrekvencia az a hely lesz, ahol a parabola csúcsa található. A módszer a 2.5. ábrán látható.



2.5. ábra: Parabola illesztése 3 pontra

A 3 mintából a középső van a legmagasabban, de a megtalált, piros ponttal jelölt csúcs még magasabban van. Mivel a jobb oldali minta nagyobb amplitúdójú, mint a bal oldali, ezért a középsőhöz képest jobbra lesz eltolva a talált csúcs.

Abban az esetben, ha a jobb és baloldali csúcs szimmetrikusan helyezkedik el és egyforma magasak, akkor az alapból megtalált csúcsnál lesz az alapfrekvencia. Ez a módszer pontosabb, mert két FFT pont között is megtalálhatjuk a pontos eredményt.

3 Matlab implementáció

Ebben a fejezetben az ECKF matlabos megvalósításáról lesz szó kissé gyakorlatiasabb megközelítésben. A Matlab környezet használatának több oka is van. A Matlab (Matrix laboratory) a mátrixok kezelésére a legegyszerűbb lehetőségeket adja. Ez azért fontos, mert a bevitt hangokat sor vagy oszlopvektorként kezeljük, a becsült jel is az lesz. Numerikus számításokban kiváló program, egy paranccsal megoldható a Fourier-transzformáció, a mátrixszorzás és egyéb műveletek. Az eredmények ábrázolása grafikusan könnyen és látványosan megoldható.

3.1 Pitch detector

Algoritmusunkban egy keret alapfrekvenciájának meghatározására használjuk a pitch detectort. Erre több esetben is szükség lesz. A Kalman-szűrő egy olyan algoritmus, amely mindig az előző iterációban számított eredmény és az adott iterációban lévő mérés alapján ad becslést az állapotváltozóra. Az első iterációban azonban nincs előző eredmény, ezért teljes mértékben egy becslésre van szükség, ez a kezdeti állapot meghatározása, melyet az 3.2.1 fejezetben tárgyalok. Abban az esetben, amikor új hang lép be és a kovariancia mátrix vissza van állítva (resetelve van), ugyanígy szükség van a kezdeti állapot meghatározására.

Már a 2.3 fejezetben is tárgyaltam az FFT fontosságát az algoritmusban. Azonban, mielőtt alkalmazzuk az FFT-t, érdemes a jelet átalakítani. Az átalakítás neve: zero padding (nulla értékekkel való kiterjesztés). A bemeneti jelsorozat végéhez nulla jelértékeket adunk, így megnöveljük a teljes jelhosszúságot. Ez úgy fogja befolyásolni a frekvenciafelbontást, hogy növeli az FFT pontok számát, amik így közelebb kerülnek egymáshoz a fekvenciatengelyen. Ezt a kiegészítést úgy tesszük meg, hogy a kettő legközelebbi hatványa legyen a mintaszám, mert így a DFT (diszkrét Fourier transzformált) gyorsabban kiszámítható. Az FFT pontok számát a kiegészítés után még meg is szorozzuk néggyel, mert így az eredmény még nagyobb felbontású lesz. Algoritmusunkban ez akkor fontos, amikor az alapfrekvencia meghatározásához csúcsokat keresünk, melynek a helye a spektrumban fogja adni a frekvenciát. Statisztikailag, ha több FFT pont van, akkor a csúcs helye nagyobb valószínűséggel lesz a valódi frekvenciánál.

3.2 Keretek feldolgozása, lépések

Algoritmusunk iteratív, ismétlődő lépéseket hajtunk végre az előre meghatározott hosszúságú kereteken. Ebből sejthetjük, hogy első lépésben fel kell osztani a bemeneti jelet keretekre. A bemeneti jel lehet mesterségesen generált, a Matlab funkcióit használva. Zenei hangot is beadhatunk, azonban azt át kell alakítani egy oszlopvektorrá, amit a Matlab hangbeolvasó függvényével tehetünk meg. Az ismétlődő lépések a kereteken a következők:

- A keretről meghatározzuk, hogy csendes keret-e (vannak-e benne harmonikusok).
- Amennyiben nem csendes, de az előző az volt vagy a számolt frekvencia eltérése meghalad egy előre beadott értéket, alapállapotba állítjuk a kovariancia mátrixot. Ezt az esetek tekintjük úgy, hogy új hang lépett be.
- Meghatározzuk a kezdeti állapotot a jel mért paraméterei alapján.
- Egy kereten, mintára-mintára végig futtatjuk az ECKF algoritmust, ami finoman végig követi a jel változásait.

A következő alfejezetekben ezeket a lépéseket tárgyalom.

3.2.1 Kezdeti állapot meghatározása

Ahogy már tárgyaltuk, a kezdeti állapot meghatározására akkor van szükség, ha egy új hang lép be és a kovariancia mátrix alapállapotba van állítva. Abban az esetben, mikor csendes keretből vált nem csendesre a hang, egyértelmű, hogy hangváltás történt. Ha viszont két egymás utáni keret nem csendes akkor nem ennyire egyszerű megállapítani a hangváltást. A pitch detector segítségével kiszámoljuk a legújabb keret alapfrekvenciáját, legyen ez f_1. Az azt megelőző keretre már van egy frekvenciabecslésünk f_0. Az eltérést(d) centekben (félhang századrésze) számoljuk a következő módon:

$$d = 1200 log_2 \left(\frac{f_1}{f_0}\right) \tag{3.1}$$

Ha megegyezik a két frekvenciaérték akkor a változás nulla, más esetben pedig egy felhasználó által beadott paraméter alapján döntjük el, hogy van-e hangváltás. Ez a paraméter a num_semitones, azt adja meg, hogy hány félhang változásnál kell új hanggal számolni. A döntést az alábbi egyenlőtlenség alapján hozzuk meg:

$$d > num\_semitones * 100 \tag{3.2}$$

Melynek teljesülése esetében hangváltással számolunk.

Amennyiben történt egy ilyen váltás, akkor a bemeneti jeltől függően érdemes kihagyni néhány keretet a frekvenciakövetésnél. Ez azért jó, mert bizonyos hangszereknél belépéskor nem annyira stabil, hanem hirtelen nagyon változik a frekvencia, ami miatt számításunk hibás lenne. A kihagyott keretek száma a felhasználótól függ, a num_buff_wait paramétertől, melyet az általam vizsgált zenei jelek esetében 1 és 3 között érdemes választani. Miután kihagytuk a megfelelő számú keretet, az algoritmus újra elkezdi számolni az alapfrekvenciát, meghatározza a kezdeti állapotot a már tárgyalt pitch detector segítségével.

3.2.2 Keret csendességének meghatározása

Fontos információ egy hangról, hogy mely időtartományban nem szólal meg. A csendes részeknél a szűrő nem ad becslést, az eredményben nem jelenik meg frekvenciaérték. Abban az esetben is így kell tennie, ha vannak zajkomponensek, feltéve, hogy maga a mérendő jel nem szólal meg.

A csendesség meghatározása keretenként történik. Különböző kritériumoknak kell eleget tenniük a kereteknek, hogy ne vegyük csendesnek őket.

Első lépésként kiszámítjuk a jel spektrális teljesítménysűrűségét (PSD-Power Spectral Density) a Welch-módszer segítségével, majd megvizsgáljuk a komponensek frekvenciáját. A felhasználó által meghatározott frekvenciahatárokon kívül eső komponenseket kiszűrjük.

Ezután a megmaradó komponensek összes energiáját kiszámítjuk az alábbi módon:

$$E = 10log_{10}\pi \sum_{i=0}^{N} \frac{PSD(i)}{N} \tag{3.3}$$

Ebben az egyenletben N a spektrális sűrűségfüggvény hossza, és csak a frekvenciaszűrés után megmaradt komponenseke összegezzük. Decibel értékben kapunk egy eredményt, amit aztán összevetünk egy felhasználó által beállított paraméterrel, az energy_threshold-dal, amelyet az általam vizsgált jeleknél -50dB-re érdemes állítani. Ha ennél kevesebb az energia, akkor csendes keretről van szó. Ez a módszer akkor eredményes, amikor nem zajos a jel, mert a zajkomponensek összege elérhetné az energiaértéket, és így helytelen következtetésre jutnánk.

Egy másik kritérium a spektrális laposság (SPF-Spectral Flatness) vizsgálata az alábbi egyenlet alapján:

$$SPF = \frac{\sqrt[K]{\prod_{i=0}^{K-1} PSD(i)}}{\frac{1}{K}\sum_{i=0}^{K-1} PSD(i)} \tag{3.4}$$

Ebben az egyenletben tehát a spektrális sűrűségfüggvény mértani közepe osztva a számtani közepével. Értéke 0 és 1 között mozoghat. Azt mutatja meg, hogy mennyire zajszerű a jel. Minél nagyobb az érték, annál zajosabb a jel. Magas értékeknél a spektrumképen a teljesítmény eloszlása viszonylag egyenletes, ami egy zajszerű jellemző. A tiszta aussi fehérzaj spektrális laposság elmélet szerint 1. (Egyetlen blokk spektrális laposság a véges számú minta miatt akkor is 1 alatt lesz, ha csak zajt tartalmaz, az elméleti 1-es értéket csak a zajblokkok átlagos spektruma közelíti.) A kapott eredményt összevetjük egy, a felhasználó által meghatározott paraméterrel, a flatness_threshold-dal, amit az általam vizsgált jeleknél 0.45-re érdemes állítani. Ha a jel spektrális laposság nagyobb ennél, akkor csendes keretről van szó. Ez a módszer zajos jeleknél lehet eredményes, mert ha nem zajos a jel, akkor nem lesz zajszerű a spektruma, alacsony lesz a spektrális laposság.

Az algoritmus mind az energiavizsgálatot mind a spektrális laposság vizsgálatot végrehajtja, így bármilyen jel-zaj viszony esetében ki tudjuk szűrni a csendes kereteket. Akkor számít egy jel nem csendesnek, ha teljesül rá, hogy az energiája elég nagy és a spektrális laposság kicsi, a paramétereknek megfelelően.

3.2.3 Kalman-szűrés megvalósítása

Az eddig leírt lépésekben keretenként vizsgáltuk a jelet. Az ECKF algoritmus viszont mintáról-mintára vizsgál, azaz a keret hosszán minden egyes mintát megvizsgál. Implementációnkban egyszerre egy kereten futtatjuk végig az algoritmust, abban az esetben, ha nem csendes a keret.

Az algoritmushoz a 2.2. fejezet egyenleteit használjuk fel. Az állapotváltozó alapállapotból indul, a kovariancia mátrix csak nullákból áll kezdetben. Az állapotváltozó alapállapota:

$$\hat{x}_0^- = \begin{bmatrix} \exp\left(2\pi j\hat{f}_0 T_s\right) \\ \hat{a}_0 \exp\left(2\pi j\hat{f}_0 T_s + j\hat{\Phi}_0\right) \\ \hat{a}_0 \exp\left(-2\pi j\hat{f}_0 T_s - j\hat{\Phi}_0\right) \end{bmatrix} \tag{3.5}$$

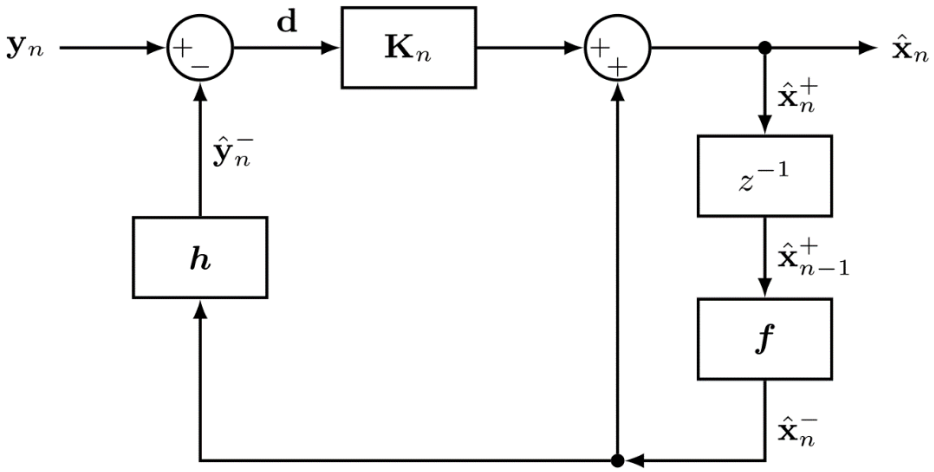
Ahol $\hat{a}_0$, $\hat{f}_0$, $\widehat{\Phi}_0$, a pitch detector által kiszámolt kezdeti értékei a változóknak. Ez alapján az érték alapján ad egy becslést a bemeneti jelre ($\hat{y}$), majd a valódi és becsült y érték alapján számol egy hibajelet. A kovariancia mátrixhoz a hiba alapján adaptív zajt adunk. Kiszámítjuk a Kalman-nyereséget a (2.8) egyenlet alapján. Ezután az állapotfrissítés meg tud történni, az a posteriori értékeket kiszámítjuk a (2.9) és a (2.11) alapján. Innentől kezdve minden elkövetkező lépésben az előző alapján tudunk állapotbecslést adni a változókra a (2.10) és (2.12) alapján. Végül kimeneti számolt értékek az állapotváltozók alapján:

$$f_1 = \frac{\ln(\hat{x}^+(1))}{2\pi j T_s}$$

$$a_1 = \sqrt{\hat{x}^+(2)\hat{x}^+(3)}$$

$$\Phi_1 = \frac{1}{2j}\ln\left(\frac{\hat{x}^+(2)}{\hat{x}^+(3)(\hat{x}^+(1))^2}\right) \tag{3.6}$$

Miután megkaptuk az eredményeket a paraméterekre egy mintánál, az algoritmus a következő mintára lép, ugyanezen lépéseket ismétli, egészen addig, amíg a blokk végére nem ér. Az 3.1. ábra ezeket a lépéseket mutatja blokkdiagrammal.



3.1. ábra: ECKF blokkdiagram

A jelölések itt ugyanazok, amiket eddig használtunk, a körök összeadást vagy kivonást jelölnek, az alsó indexben a diszkrét időpont van. A z^{-1}-el való szorzás pedig egy késleltetést jelent. A h betű jelentésére a magyarázat a következő: A (2.6) egyenlet szerint y=Hx, ami általános esetben y=h(x). Az y=h(x) lineáris megfigyelés esetében egyszerűsödik y=Hx-re.

3.2.4 A változtatható paraméterek szerepe

Ebben a fejezetben a felhasználó által beállítható paramétereket foglalom táblázatba, névvel, szereppel, jelöléssel. Ezen paraméterek megfelelő beállítása nagyon fontos ahhoz, hogy optimálisan működjön az algoritmus, helyes eredményeket kapjunk.

Jelölés	Leírás
y	Oszlopvektorként tárolt jel, aminek a frekvenciáját követjük
num_buff_wait	Meghatározza, hogy hány keretet kell kihagyni abban az esetben, ha új hang lép be. Ha erősebb a hangszer belépése, magasabbra kell állítani, a vizsgált esetekben 1 és 3 közötti értékek voltak megfelelőek.
num_semitones	Meghatározza azt, hogy hány félhang különbségnek kell lennie két egymás utáni keret között, hogy új belépő hanggal számoljunk. Általában 2, ami egy egész hang különbséget jelöl.
system_noise_coeff	Ez a zajtényező, azt határozza meg, hogy mennyire illeszt a szűrő a modellre, szemben a mért értékekkel. Nagyobb együttható esetén a zaj szűrése hatásosabb, de a szűrő lassabban áll be. A vizsgált esetekben 7 és 11 közötti az értékek voltak megfelelőek.
bs	Blokkméret, egy keretben található minták számát határozza meg. Ha kisebb a blokkméret, akkor gyorsabban lehet követni a frekvencia változását, viszont nagyobb hibával. Mindig 2 hatványa, vizsgálatainkban 1024 vagy 2048.
num_peaks	A pitch detector ennyi csúcs alapján számítja ki a kezdeti frekvenciát. Vizsgált eseteinben 2 és 6 közötti értékeinél kaptuk a legjobb eredményeket.
energy_threshold	Az a minimális energia decibelben, amely alatt egy keret csendesnek van számolva. Jelen algoritmusban -50 dB.
flatness_threshold	Az a maximális spektrális laposság érték, ami felett egy keret csendesnek van számolva. Jelen algoritmusban 0.45.
frequency_limits	Azon frekvenciahatárok, amelyeken kívüli frekvenciájú komponenseket elvetünk. A bemeneti jeltől függ az érték, zaj meghatározásához is fontos szerepe van.

peak_threshold	A pitch detector azon csúcsokat tartja meg, amelyek a legmagasabb csúcsnál maximum ennyi decibellel kisebb amplitúdójúak. A vizsgált esetekben -30 és -40 dB közötti értékekre van állítva.

3.1. táblázat: felhasználó által megválasztható paraméterek

3.3 Több jel követése

Ebben a fejezetben annak a feladatnak a megoldását írom le, hogy az alapjel mellett a szűrő kövessen egy másik komponenst is. Alapvetően a módszer ugyanaz marad, de a dolgozat során említett lépések implementációja kissé megváltozott.

A pitch detector, mellyel egy keret alapfrekvenciáját határoztuk meg, csak annyiban változott, hogy az elején a frequency_limits dimenziójától függően megvizsgálja, hogy hány komponenst követünk. Használata az algoritmusban szintén megegyezik az eredetivel.

A csendes keretek meghatározásában a kezdeti különbség szintén ez. Ha több komponens van az azt jelenti, hogy több frekvenciahatár van megadva, és keretenként a külön komponenseket más határokon szűri. A függvény annyi dimenziós eredményt ad vissza, ahány komponens van és a különböző komponensek egymástól függetlenül lehetnek csendesek vagy sem.

A szűrés megvalósításához is módosítani, bővíteni kell az algoritmust. A bemeneti jel(y) dimenziója nem fog változni, viszont a változó igen. A frequency_limits dimenziója alapján megkapott komponens szám alapján bővítjük az állapotvektort, megfigyelési mátrixot, kovariancia mátrixot, Jacobi-mátrixot. Ez a bővítés a 3 tagú oszlopvektorok esetén azt jelenti, hogy annyiszor 3 tagú oszlopvektorok lesznek, ahány komponens van. Az alapesetben 3x3as mátrixok (pl. kovariancia mátrix) esetén $3n \times 3n$ típusú mátrixok lesznek, ahol n a komponensek száma.

A bemeneti jelet még mindig keretenként vizsgáljuk. Minden keretnél megállapítjuk, hogy megjelent-e új hang. Ez különbözhet több komponens szempontjából, mert egy keret lehet csendes az egyik frekvenciahatárain belül, de nem csendes egy másikénál, ezért végig vizsgáljuk a keretet komponensenként. Azt, hogy a szűrőt mikor kell alapállapotba állítani, azaz a kovariancia mátrix mely részét kell nullába állítani egy külön logikával állapítjuk meg. Amennyiben voltak komponensek, de nem volt a frekvenciájukban egy megadott paraméternél nagyobb változás, vagy csendes keretek voltak egymás után, nincs dolgunk. Ha valahol lépett

be új hang, akkor az állapotvektor azon 3 elemét állítjuk alapállapotba, amelyekhez az adott komponens állapotváltozói tartoznak, pl. első komponens: f_1, a_1, Φ_1, második komponens: f_2, a_2, Φ_2. Mivel tudjuk, hogy az első komponens változói mindig az első három sorba kerülnek a vektorban, a másodiké a második háromba és így tovább, ezért a megfelelő indexű elemeket tudjuk alapállapotba állítani. A kovariancia mátrixnál is hasonlóan történik, csak annyi különbséggel, hogy 3 sort majd 3 oszlopot kell nullázni az indexeknek megfelelően.

Az ECKF futtatásához egy kereten szintén egy külön logikával kell megállapítani. Amennyiben egy kereten belül egy komponens nem csendes és volt új hang belépés, úgy az a komponens kiválasztásra kerül ahhoz, hogy az ECKF algoritmust végigfuttassuk rajta. Ha egyik komponens sincs kiválasztva a keretben, akkor átugorjuk a szűrést. Ezután az 3.2.3.-ban leírtaknak megfelelően végig futtatjuk a kiválasztott komponensen, a megfelelő mátrix elemeken. Előfordulhat az is, hogy több komponens is ki van választva, olyan esetben annyiszoros számú elemen fut le az algoritmus, ahány komponens ki van választva. Amennyiben egy komponensről azt feltételezzük, hogy csendes, akkor nem számolunk vele a hibajelben (y-Hx). Ez megváltoztatja a visszacsatolást a nem csendes komponensektől függően.

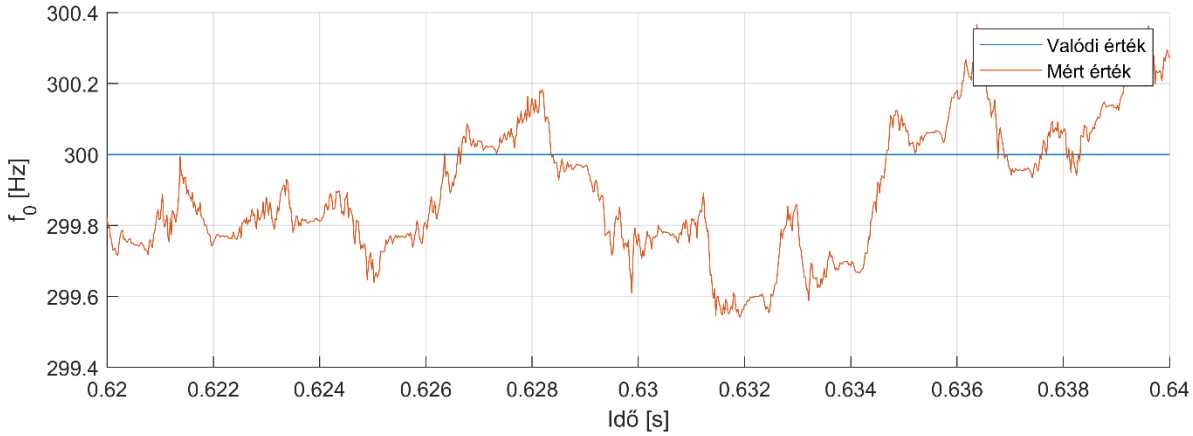
4 Tesztelés

Ebben a fejezetben az algoritmus különböző bemeneti jelekkel való tesztelését mutatom be. Először a mesterséges jeleket, majd a zenei jeleket, amiket össze is hasonlítok egy másik algoritmussal, végül pedig a több komponens detekcióját vizsgálom.

4.1 Mesterséges jelek

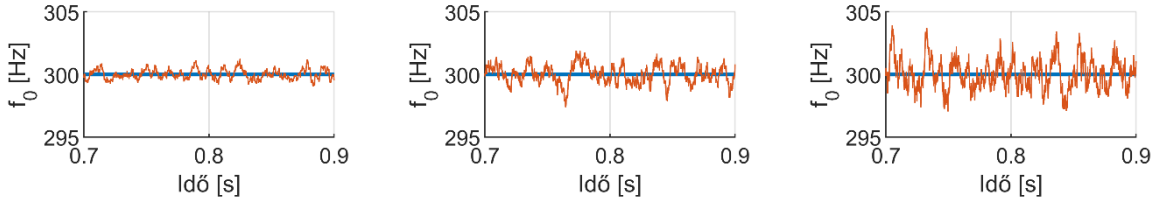
A mesterséges jelek tesztelése fontos információkat szolgáltat arról, hogy az algoritmusunk hogyan reagál a bemeneti jel bizonyos paramétereinek változására, legyen az nagy zaj, a szinuszjel modulációja vagy megszakítása. A mesterséges jeleket a Matlab függvényei segítségével hozunk létre.

Az első jel, amit teszteltem, a legalapvetőbb, a szinuszjel. Az algoritmus zajtűrését bemutatva különböző SNR-ek (jel-zaj viszony) mellett mutatom be az eredményt. Kék vonal jelöli a valódi értéket, narancssárga azt, mit az ECKF-fel mérünk. Az alapfrekvencia jelölése f_0.



4.1. ábra: Szinuszjel alapfrekvenciája (f=300Hz, SNR=20dB)

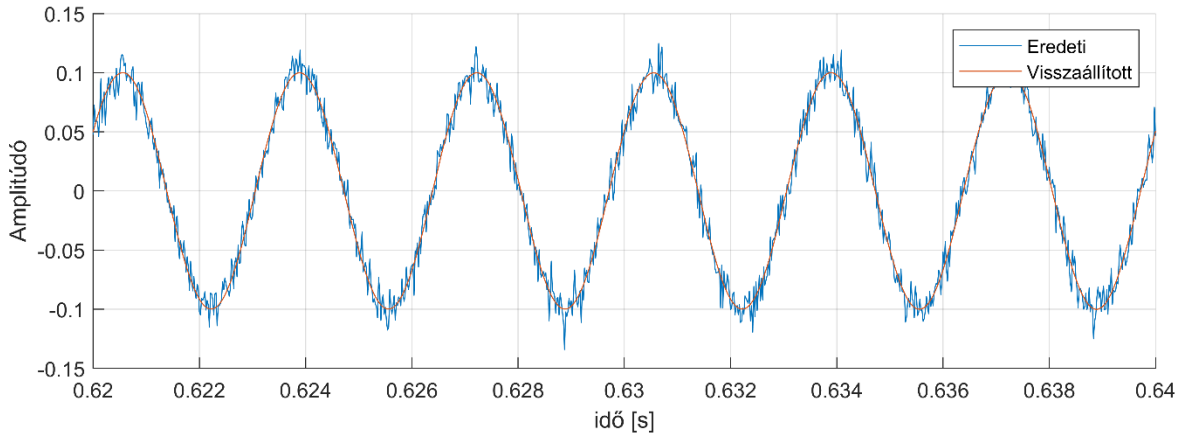
Jó SNR esetén a szűrő nagyon pontos eredményt tud adni, az alapfrekvenciától csupán néhány tized Hertzzel tér el a mért eredmény. Nézzük meg azokat az esetet, amikor nagyobb zajjal van terhelve a jel:



4.2.ábra: Szinuszjel alapfrekvenciája (f=300Hz)

A bal oldali esetben az SNR 15dB, a frekvenciaértékek varianciája 0.1202. A középső esetben az SNR 10dB, és a szűrő nagyobbat téved, mint a bal oldali esetben, viszont így sem megy a tévedése 3 Hertz fölé. A jobb oldali ábrán az SNR 5dB és hasonló a helyzet, kicsit megnövekszik a tévedés átlagos nagysága. Abban az esetben is, amikor az SNR-t 5 dB-re csökkentettük, a kapott eredmény így is maximum 4 Hertzzel tér el az eredetitől.

Érdemes megvizsgálni azt is, hogy a kapott alapfrekvencia értékek alapján hogyan is néz ki a visszaállított jel. Mivel az ECKF nem csak az alapfrekvenciát, hanem az amplitúdót és a fázist is követi, ábrázolni tudjuk a jelet.

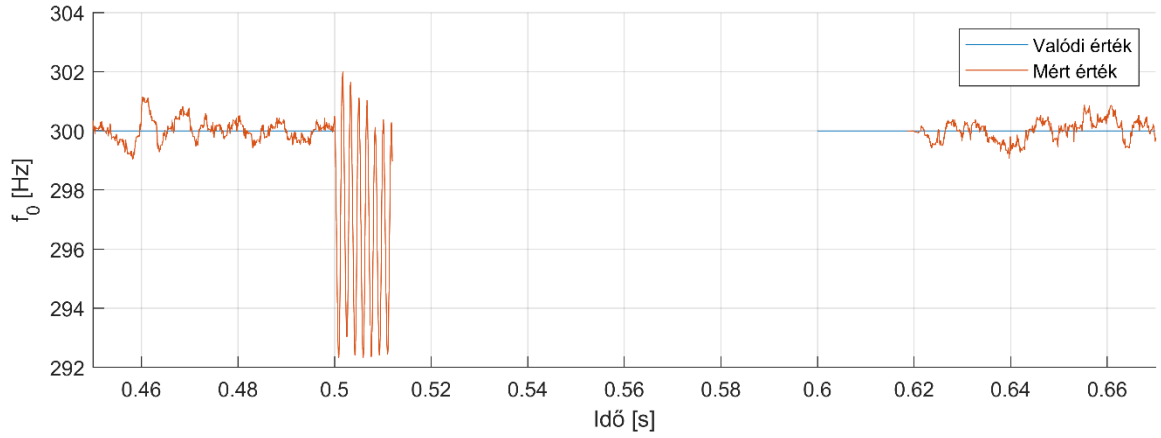


4.3.ábra: Visszaállított szinuszjel (f=300Hz, SNR=20dB, a=0.1)

A visszaállított jel szinte teljesen megegyezik azzal, mintha generáltunk volna egy 300 Hertzes szinuszjelt, mutatva, hogy az algoritmus a jel minden fontos paraméterét pontosan követi.

A következő vizsgált jel a megszakított szinuszjel. A megszakítás azt jelenti, hogy egy adott ideig sima szinuszjelt generálunk, majd azt követi egy olyan rész, ahol nincs szinuszos

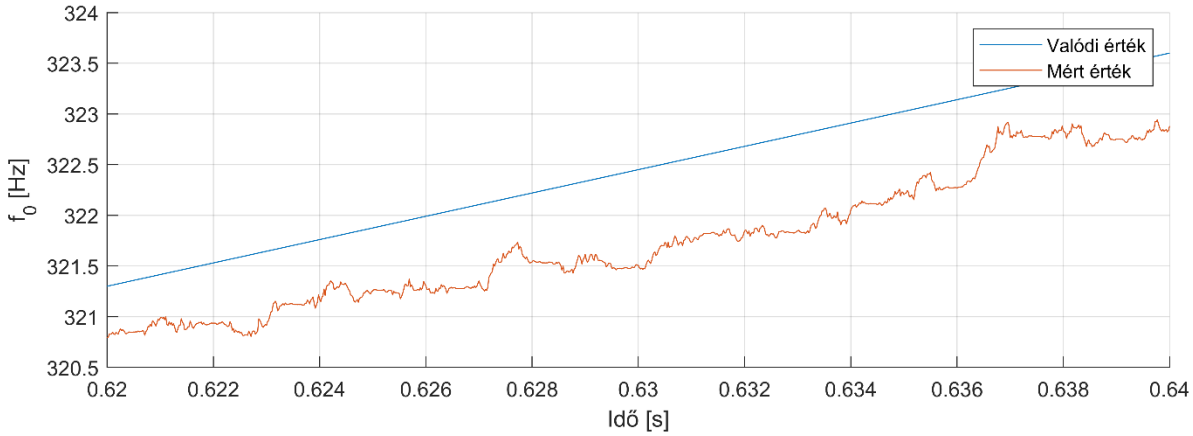
komponens, és ez ismétlődik. Ennek a jelnek a vizsgálata azért fontos, mert csendes keretek kezelését tudjuk vizsgálni, meg azt is, hogy mennyi idő alatt alkalmazkodik a szűrő egy bejövő hanghoz.



4.4. ábra: Megszakított szinuszjel alapfrekvenciája (f=300Hz, SNR=20dB)

Amíg sima folytonos szinuszjel van, az eredmény megegyezik a 4.1. ábrával. A hirtelen csendes résznél az algoritmusnak szüksége van kb. egy századmásodpercre ahhoz, hogy csendesként mutassa a jelet. Egy rövid részen a szűrő nem tudja eldönteni, hogy folytatódik-e a 300 Hertzes jel vagy már csendes résznél van-e, ennek oka a jel keretekre osztása és a beállási idő. Amikor a hang újra megszólal, szintén egy késést figyelhetünk meg, mert amikor új hang lép be, a szűrő kihagy néhány keretet és csak a stabil, állandó állapot beállásánál méri tovább a szokásos módon a jelet.

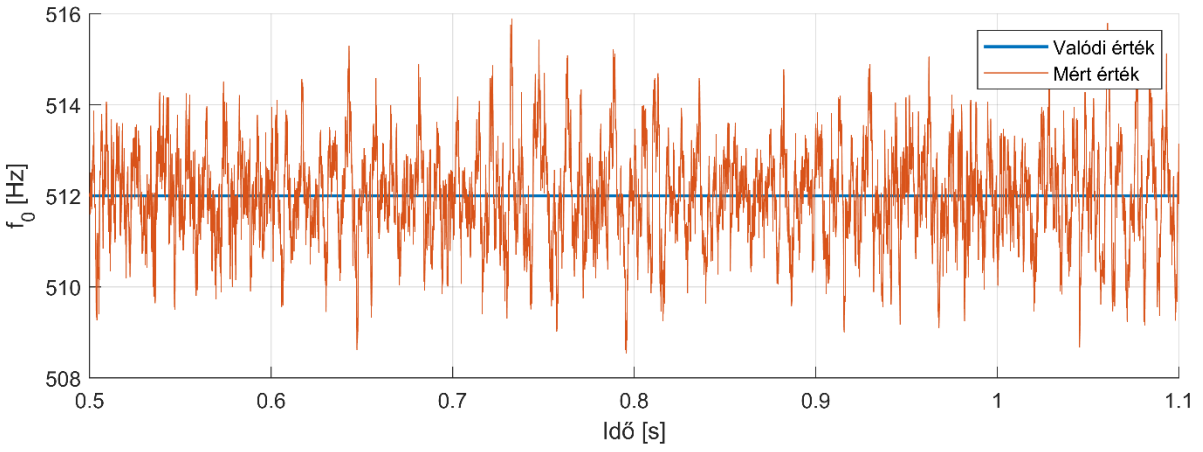
A megszakított szinuszjellel a csendes keretek és az azután belépő hangok követését teszteltük. Új hang viszont úgy is létrejöhet, hogy már vannak meglévő komponensek, viszont az azt követők alapfrekvenciája annyira más értékű, hogy másik hangként kell vizsgálunk őket. Erre jó példa a változó frekvenciájú szinuszjel. Jelen esetben lineárisan növekvő frekvenciájú szinuszjelt vizsgálunk.



4.5.ábra: Lineárisan növekvő frekvenciájú szinuszjel alapfrekvenciája (f_0=250Hz, f_1=480Hz, SNR=20dB)

A kiindulási frekvenciát f_0-val, a végfrekvenciát f_1-gyel jelöltük, de a képen a kapott eredménynek csak egy kis részletét ábrázoltam, hogy jobban látható legyen a becslés pontossága. A szűrő egy kis késéssel, de tudja követni a folyamatosan növekedő frekvenciát, 1-2 Hertzzel kisebb értékeket mutat, mivel mindig az előző beállt állapot alapján számol.

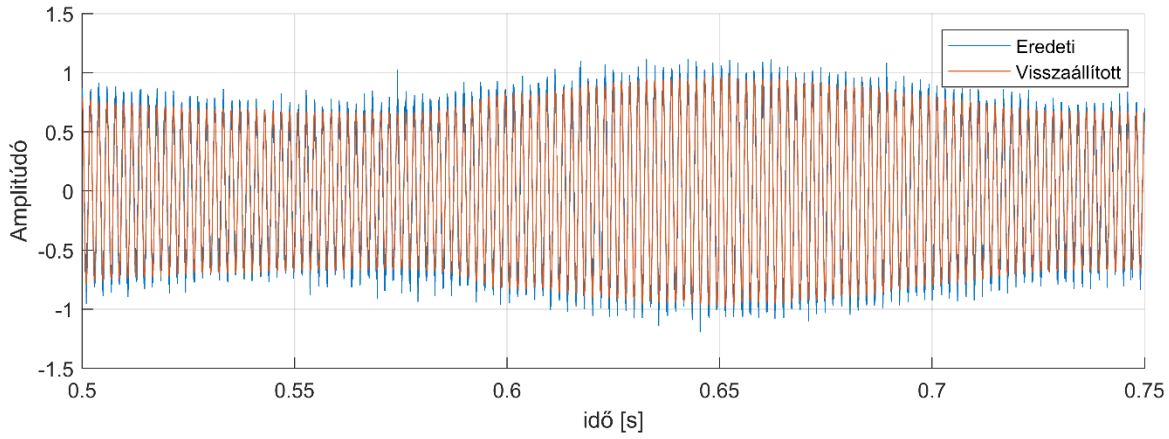
Habár az amplitúdómoduláció során a jel frekvenciáját nem változtatjuk, mégis érdekes lehet számunkra, hogy az ECKF hogy viszonyul hozzá. Ennél a modulációnál az információt az amplitúdó változása hordozza, így fontos megnézni, hogy a szűrő tudja-e követni ezt.



4.6. ábra: AM jel alapfrekvenciája (f=512Hz, SNR=20dB, f_m=5Hz)

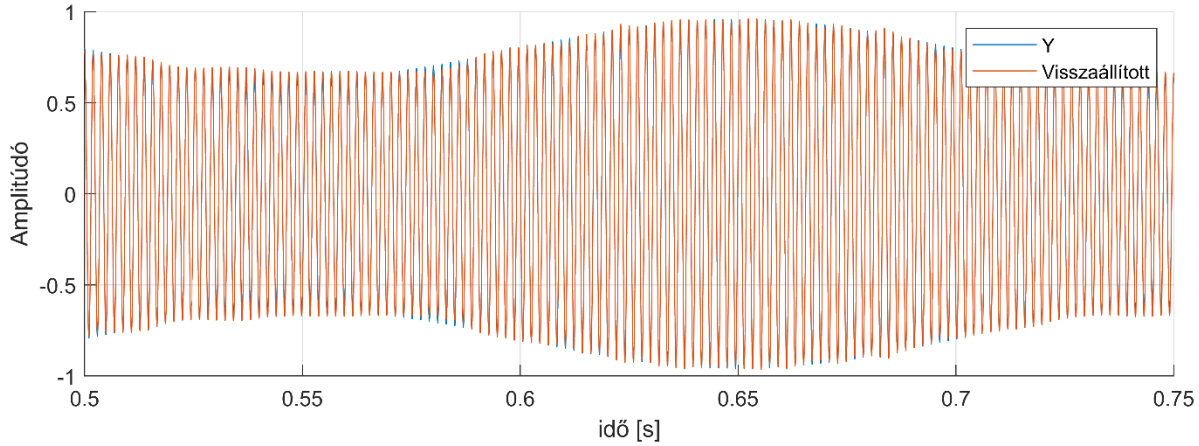
Itt f_m a moduláló frekvenciáját jelenti. Az eredmény hasonló a szinuszjelnél tapasztaltakhoz, viszont ugyanolyan SNR mellett sokkal nagyobb a tévedés a frekvenciában,

mert az amplitúdó változik és a paraméterek összefüggenek az algoritmusban. Nézzük meg, hogy mennyire pontosan tudjuk visszaállítani az eredeti jelet.



4.7. ábra: Visszaállított AM jel (f=512Hz, a=0.8, SNR=20dB, f_m=5Hz, a_m=0.2)

Látható, hogy a visszaállított jel amplitúdója kisebb, mint az eredetié, ennek az az oka, hogy az eredeti jelhez hozzáadódik a zaj, ami megnöveli az amplitúdóját, a visszaállított jelnél viszont a szűrő biztosított zajszűrést. Emiatt megvizsgálom az eredeti zaj nélküli jel viszonyulását a szűrő zajos jelhez adott eredményéhez a 4.8. ábrán.

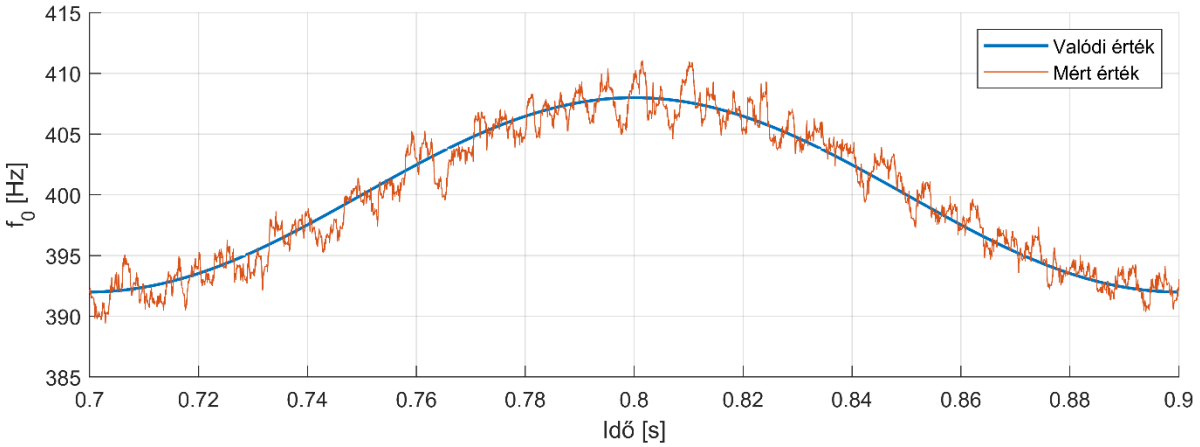


4.8. ábra: Visszaállított AM jel (f=512Hz, a=0.8, SNR=20dB, f_m=5Hz, a_m=0.2)

A visszaállított jel és az eredeti, zajszennyezés előtti jel amplitúdója közötti eltérés csupán század nagyságrendű. Ez az ECKF zajszűrésének működését támasztja alá.

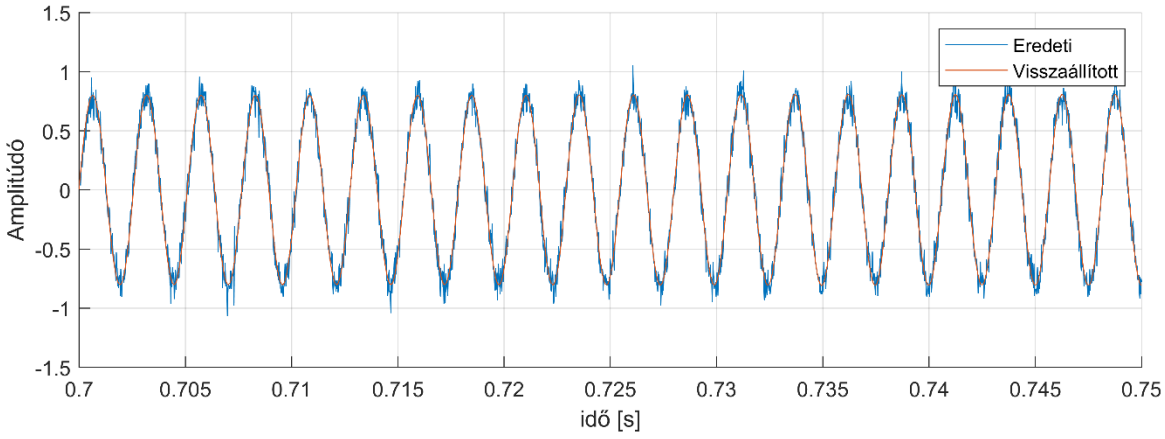
A frekvenciamoduláció során a jel frekvenciájának megváltoztatása hordozza az információt. A vivőjel frekvenciája és a moduláló jel amplitúdója között van kapcsolat. A

hirtelen frekvenciaváltások követése nagyon fontos ennél a jelnél. A 4.9. ábrán látható a frekvenciakövetés eredménye az FM jelnél.



4.9. ábra: FM jel alapfrekvenciája (f=400Hz, SNR=20dB, f_m=0.5)

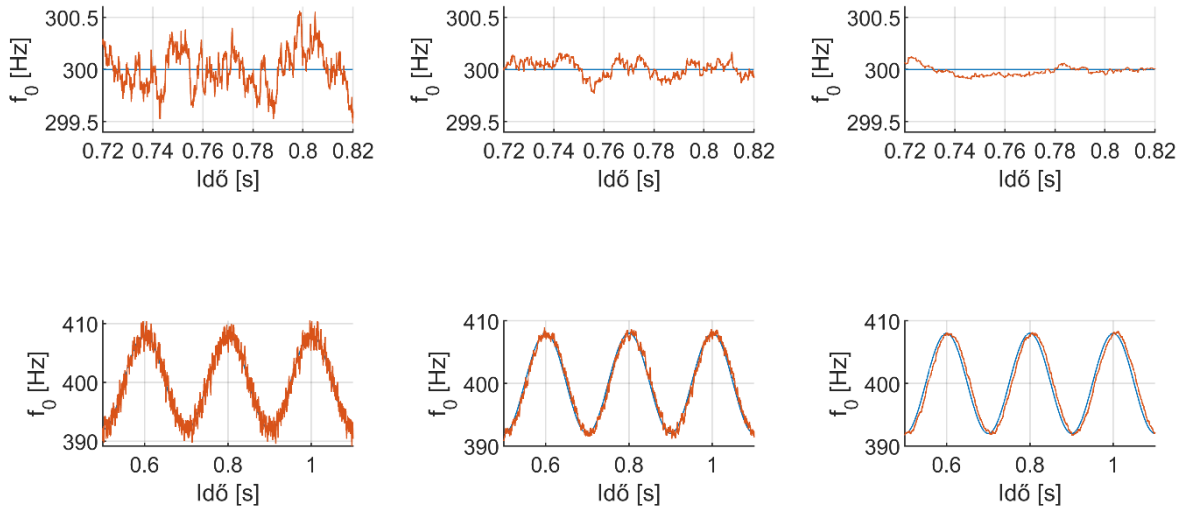
Látható, hogy mért érték maximum néhány Hertzzel tér el a valódi értéktől, hasonló mértékben, mint a szinuszjelnél. A 4.10. ábra mutatja, hogy az eredeti jelhez képest mennyire pontosan sikerült visszaállítani az FM jelet.



4.10. ábra: Visszaállított FM jel (f=400Hz, a=0.8, SNR=20dB, f_m=0.5, a_m=0.02)

A kapott eredmény hasonló a szinuszjelnél tapasztaltakhoz.

A felhasználó által beadott paraméterek már tárgyalásra kerültek az 3.2.4. alfejezetben. A zajtényező (system_noise_coeff) hatása nagyon fontos az algoritmus szempontjából. Növelésével a zaj szűrése jobb lesz, viszont lelassul a szűrő stabil állapotra való beállása. A hatást a szinuszjelen és az FM jelen mutatom be.



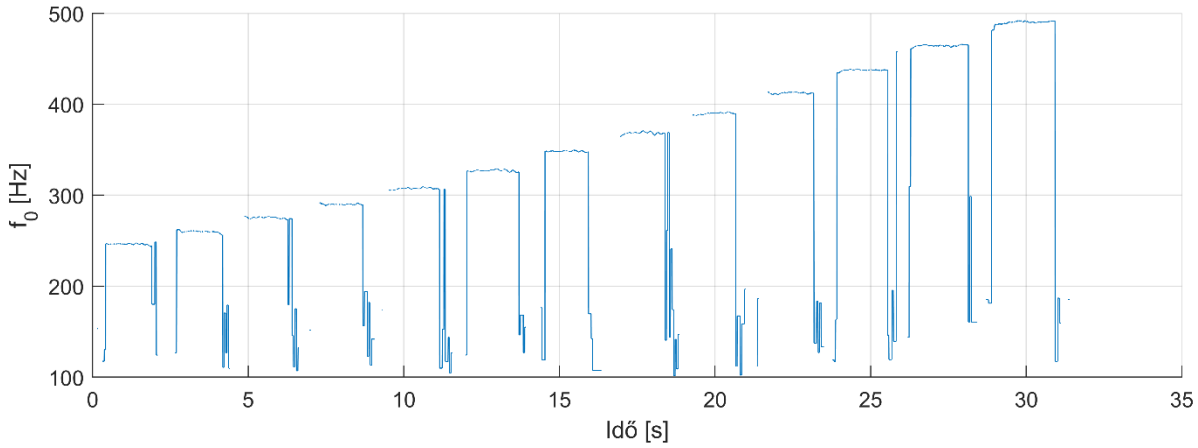
4.11. ábra: Szinuszjel (f=300Hz, SNR=20dB) és FM jel (f=400Hz, SNR=20dB, f_m=5)

A felső 3 mérés a szinuszjel alapfrekvenciájára, az alsó 3 az FM jel alapfrekvenciájára vonatkozik. A zajtényező különbözik az oszlopokban csak: bal oldalt 7, középen 8, jobb oldalt pedig 9. A szinuszos jelnél a kapott frekvenciaértékek varianciája a valódi jelnél természetesen 0, a mért jelnél a zajtényező 7es értékénél 0.0458, 8as értékénél 0.0091, 9es értékénél pedig csak 0.0032, tehát matematikailag belátható a jobb zajszűrés. A variancia számításnál a csendes részeket nem vettük figyelembe. Az ábrákról is látható, hogy a zajtényező növelése pontosabb eredményt ad a szinuszjelnél, az FM jelnél is kevésbé zajosabb eredményt mutat. A zajtényező növelésének hátrányát is bemutatják az FM jeles ábrák, a nagyobb értékénél a mért eredmény jobban késleltetve van az eredetihez képest.

4.2 Zenei jelek

Zenei jelek tesztelése azért hasznos, mert információt szolgáltat az algoritmus való életben való felhasználásáról. Ebben az alfejezetben különböző hangszereken tesztelem az algoritmust, és össze is hasonlítom egy másik népszerű frekvenciamérő algoritmussal.

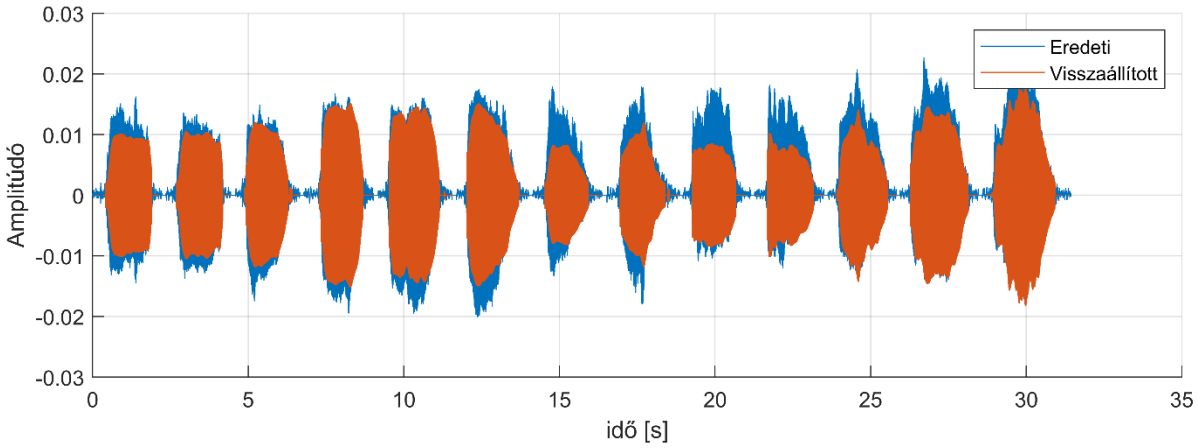
Elsőként egy fuvola hangján futtatjuk az algoritmust. Egy viszonylag hosszabb jelet vizsgálunk, aminél több hangot is megszólaltatnak, csendes részek is vannak benne. Az eredmény a 4.12. ábrán látható.



4.12. ábra: Fuvola hang alapfrekvenciája

Látható az ábrán, hogy mely időpontokban szólal meg a hangszer és mikor nincs hang. Az első hang frekvenciája 245 Hertz, utána mindegyik $2^{1/12}$-szerese, azaz egy félhanggal van feljebb a kromatikus hangsorban. Az utolsó hang frekvenciája 490 Hertz, ami egy oktávval van feljebb az elsőnél, azaz kétszeres a frekvenciája, így a teljes kromatikus hangsort lefedi a jel. Látható, hogy amikor megszólal egy hang, van egy kis beállási idő, ahol rossz eredményt ad a szűrő, szintúgy a hangok elhalkulásánál, de a keretek nagy részében pontos eredményt kapunk.

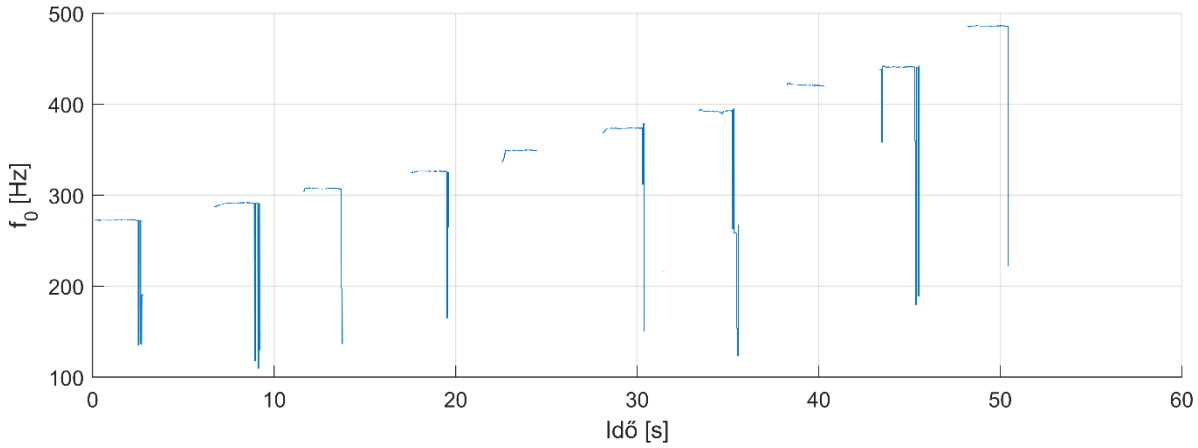
Ahogy a mesterséges jeleknél is hasznosnak bizonyult, itt is vizsgáljuk meg, hogy az eredeti jel és a szűrő által kiadott eredményekből visszaállított jel mennyire hasonló (4.13. ábra).



4.13. ábra: Visszaállított fuvola hang

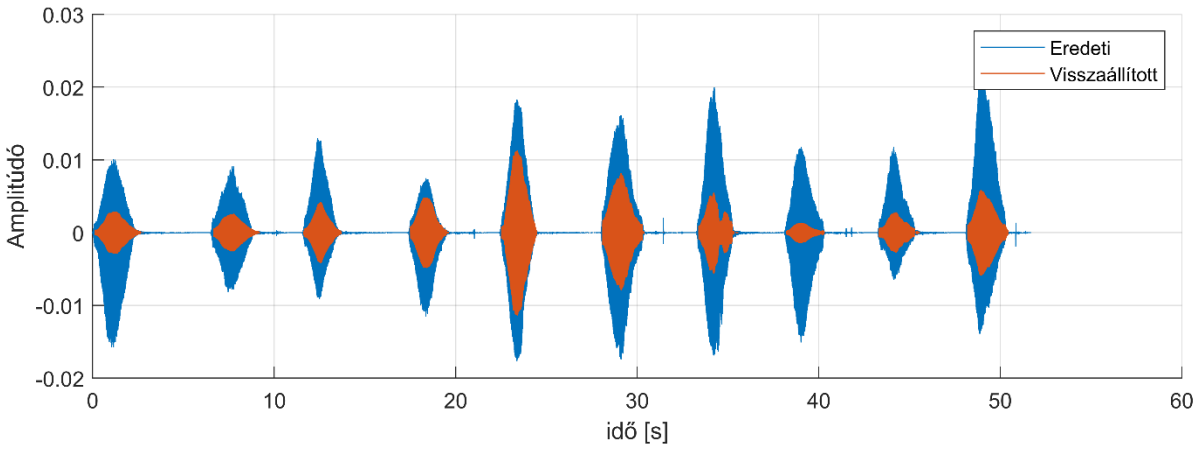
Látható, hogy a visszaállított jel megfeleltethető az eredetinek, kisebb különbségek adódnak az amplitúdóban, de ez nem probléma, mert a szűrő csak az alapharmonikust állítja vissza, így, ha vannak nagy amplitúdójú felharmonikusok, akkor nem fog egyezni a visszaállított és az eredeti jel. Algoritmusunk ugyanabban az időtartamban mutatja hangosnak a jelet, mint az eredeti, tehát közel valódi időben tudta követni.

A következő ábrán ugyanilyen mérést végzünk el, de egy brácsa hang frekvenciáját vizsgáljuk, hogy megbizonyosodjunk arról, hogy más hangszerrel is működik az algoritmus.



4.14. ábra: Brácsa hang alapfrekvenciája

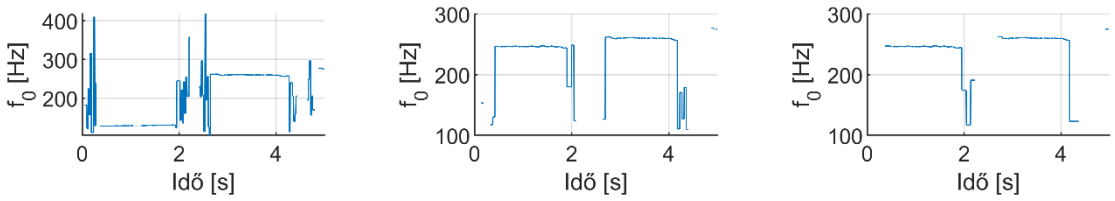
Ennél a jelnél is egy félhang a lépésköz a megszólaló hangok között, habár nem megy el a kétszeres frekvenciáig. Az eredmény ennél a hangszernél is pontos.



4.15. ábra: Visszaállított brácsa hang

A visszaállított jel a 4.13. ábrához hasonlóan itt is megfeleltethető az eredetinek.

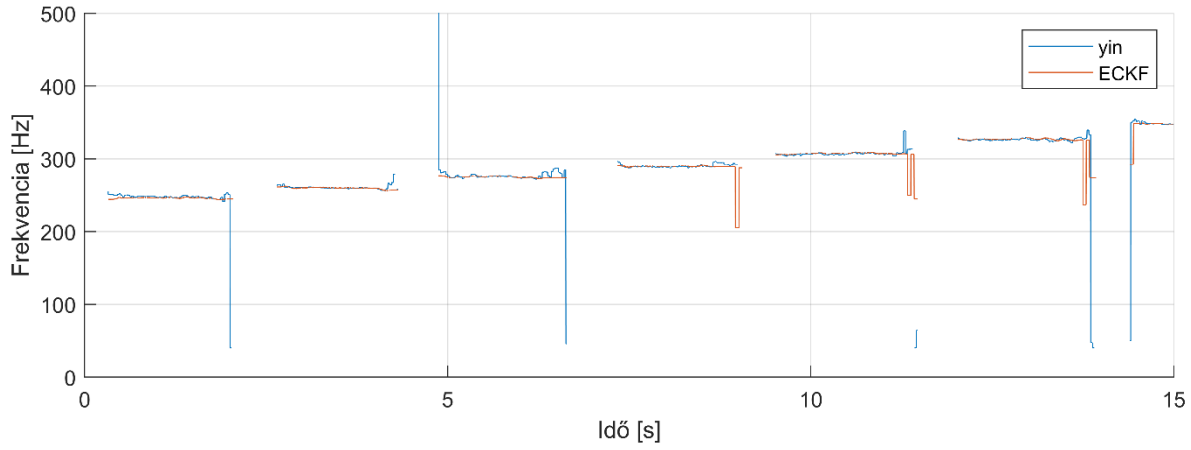
A blokkméret egy fontos, felhasználó által meghatározható paraméter, melynek csökkentésével gyorsabban tudjuk követni az alapfrekvencia változását, de pontatlanabbul tudjuk követni a jelet. A 4.16. ábra három különböző blokkméret esetén kapott eredményeket mutatja, a vizsgált jel a furulya hangja.



4.16. ábra: Furulya hang alapfrekvenciája különböző blokkméreteknél

Az ábrán a bal oldali esetben a blokkméret 1024, középsőnél 2048, jobb oldalinál 4096. Látható, hogy bár a kisebb, 1024es blokkméretnél a hirtelen változásokat jobban tudjuk követni, a túl kicsi ablak miatt nem tud időben becslést adni az alapfrekvenciára a szűrő és így az első hangra teljesen hibás eredményt ad. A középső a zenei jeleknél általában optimális eset, van egy kis beállási idő hangváltáskor, de alapvetően, mint már tisztáztuk, jó eredményt ad. A jobb oldali esetben a nagyobb blokkméret miatt már később tud reagálni az alapfrekvencia változására a szűrő, így kissé tovább tart meg rossz eredményt. Érdekes meggondolni, hogy maga az algoritmus lefutására szükséges idő hogy függ a blokkmérettől. Mindegyik mérésnél megadok egy olyan arányszámot, ami a lefutáshoz szükséges idő és a jel időhosszának hányadosa. A bal oldali esetben ez 1.16, a középsőnél 0.95, a jobb oldalinál 0.76. Ez azt jelenti, hogy mivel az utóbbi kettőnél 1 alatt van a szám, real-time sikerült végrehajtani a mérést, ami nagyon fontos.

Kitűzött feladatunk volt az ECKF-n alapuló algoritmusunkat más frekvenciamérő módszerrel összehasonlítani. A másik módszer, aminek vizsgáljuk az eredményét a YIN frekvenciabecslő lesz [5]. A YIN módszer évekkel előbb jelent meg, mint az ECKF, és bár céljuk ugyanaz, módszerük nem egyezik meg. Mind a két módszernél a jelet keretekre bontjuk, de a YIN a frekvenciabecslést egy egész keretre adja szemben az ECKF-rel, amely egy kereten belül mintáról mintára ad becslést. Kevesebb felhasználó által beállítható paraméterrel rendelkezik a YIN és a lépések is mások benne. Időtartománybeli módszere az autokorrelációs módszer, amelynek alkalmazása után hibacsökkentő lépéseket tesz. Nézzük meg, hogy a jól ismert furulyás jelünknél milyen eredményt kapunk az ECKF-rel összehasonlítva:



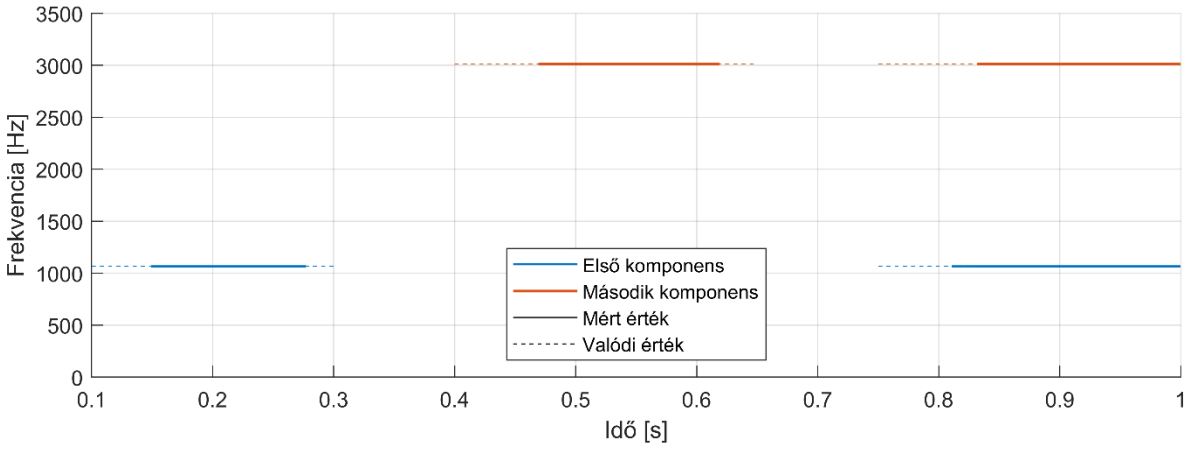
4.17. ábra: Furulya hang alapfrekvenciája YIN és ECKF algoritmusokkal mérve

A YIN algoritmus is viszonylag jó eredményt ad az alapfrekvenciára, de amikor alá vagy fölébecsli az alapfrekvenciát, akkor jóval nagyobbat téved, mint az ECKF. Az is látszik, hogy az ECKF által adott becslés sokkal simább, a vonala egyenesebb, kevesebb a kitérés egy-egy hangnál, ahol elméletben állandó frekvenciát kéne mutatnia a szűrőknek. Összeségében az ECKF jobban kezeli a zenei jelnél a változást, kevésbé hajlamos nagyban eltérni a valóságtól.

4.3 Több komponens követése

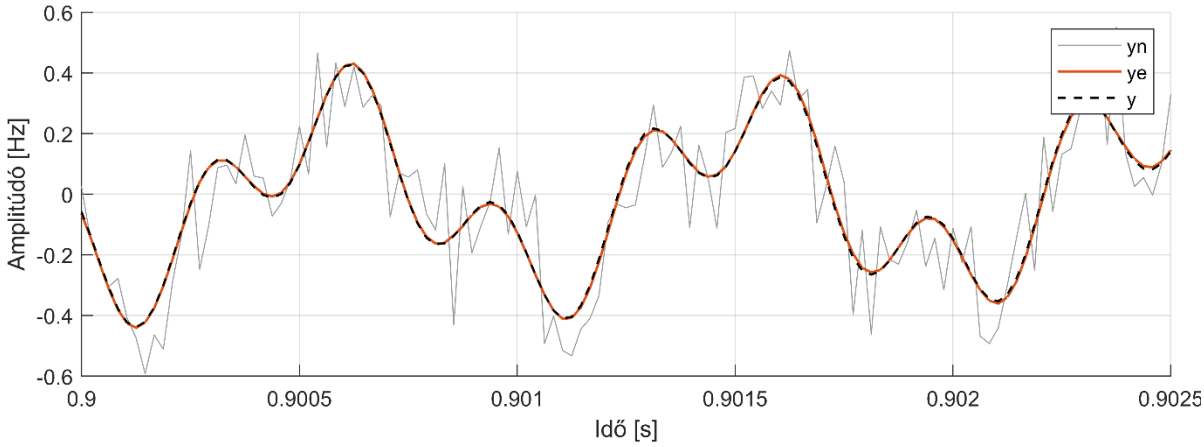
Ebben az alfejezetben a Kalman-szűrő azon kiterjesztését tesztelem, amely lehetővé teszi, hogy egy jelnek több különböző frekvenciájú komponensét vizsgáljam egy időben. Ez leginkább akkor hasznos, amikor a két komponens jól behatárolható, frekvenciájuk között jelentős eltérés van. Ahhoz, hogy az algoritmus működjön, a frekvenciahatárokat előre be kell állítani, lehetőleg úgy, hogy ne fedjék egymást.

A megvizsgált jelünk úgy épül fel, hogy két különböző frekvenciájú komponenst tartalmaz. Először csak az egyik jelenik meg, utána csak a másik, végül mind a kettő egyszerre, hogy minden esetet tesztelni tudjunk. Természetesen a jelhez hozzáadunk zajt is.



4.18. ábra: Több komponens alapfrekvenciája (f1=1067, f2=3013)

Látható, hogy itt is van egy kezdeti beállási idő, amikor még az algoritmus nem mutatja megjelenő komponens alapfrekvenciáját, de utána jól beáll a megfelelő értékre. A szűrő mind a három esetben meg tudja különböztetni a komponenseket és a várt frekvenciaértéket mutatja. Komponensenként jól működik a szűrő csendes keret detekciója. A visszaállított jelet is vizsgáljuk a 4.19 ábrán.

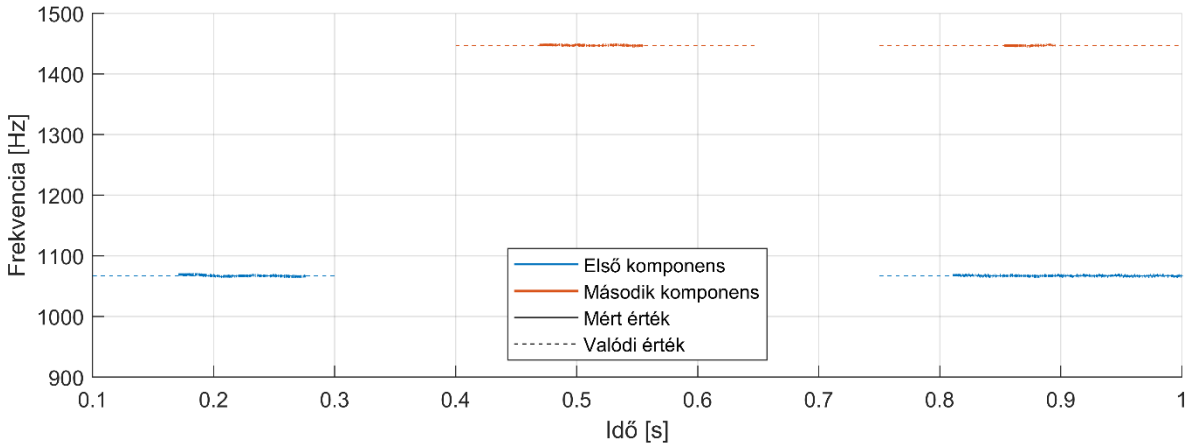


4.19. ábra: Visszaállított többkomponensű jel

A 4.19. ábrán az yn jelöli az eredeti jelet zajterheléssel, y az eredeti jel zaj nélkül, ye pedig a szűrő által visszaállított jel. Y és ye majdnem teljesen megegyezik, tehát a szűrő több komponens esetében is pontosan méri mind a frekvenciát, mind az amplitúdót és a fázist.

A következő ábrán ugyanilyen jelen fogom megvizsgálni az algoritmus működését, úgy, hogy csak a frekvenciaértékek különböznek. Ahhoz, hogy az algoritmus meg tudja

különböztetni a komponenseket, a frekvenciahatárokat is át kellett állítanom. A 4.18. ábrán az 1067 Hertzes komponenst 100 és 2000 Hertzes határok, a 3013 Hertzes komponenst 2500 és 5000 Hertzes határok között kerestük, most az 1067 Hertzes komponenst csak 100 és 1200 Hertz között, az 1447 Hertzest pedig 1300 és 2000 Hertz között keressük. Az eredmény 4.20. ábrán látható.



4.20. ábra: Több komponens alapfrekvenciája (f1=1067, f2=1447)

Látható, hogy amikor csak az egyik komponens jelenik meg, akkor, ha kissé pontatlanabbul is, mint a 4.18. ábrán, de tudja követni a szűrő a frekvenciát. A probléma akkor adódik, amikor a két komponens egyszerre jelenik meg, eleinte a szűrő érzékeli mind a kettőt, de rövidesen egy komponensként kezeli a jelet. Ez hibás és mutatja, hogy az algoritmusnak vannak korlátai, jelen esetben a két komponens frekvenciája között szükséges minimális különbség.

5 Összefoglalás

A dolgozat célja a frekvenciamérés és -követés megvalósítása Kalman-szűrővel és annak továbbfejlesztése volt.

Először a Kalman-szűrő működésének leírásához végeztem irodalomkutatást. Ehhez felhasználtam olyan cikkeket, amelyek az ECKF megvalósításáról, működéséről szólnak. Természetesen online forrásokra is támaszkodtam.

Bár tanulmányaim során már kellett foglalkoznom a Matlab programnyelvvel, jelen dolgozatom céljainak megvalósításához sokkal jobban meg kellett tanulnom használni azt. Az algoritmus megvalósításához nagy segítség volt egy már meglévő ECKF implementáció [6]. Ezt az implementációt fejlesztettem tovább, hogy átláthatóbb, stabilabban működő legyen. A felhasználó által megválasztható paraméterek használatát is egyszerűbbé, optimálisabbá tettem. Miután elkészült a működő implementáció, kiterjesztettem a szűrőt több komponens frekvenciájának egyidejű követésére is.

A létrehozott implementációt teszteltem először különböző mesterséges jeleken, majd zenei jeleken is. A zenei jeleket szintén a Matlab segítségével alakítottam át feldolgozható bejövő jellé. A szűrő minden esetben jó eredményeket adott ki, még zajos bejövő jelek esetében is. Utánanéztem egy másik frekvenciakövető algoritmusnak, a YIN módszernek, össze is hasonlítottam a működését az ECKF-rel. Vizsgáltam olyan jelet is, melynek két, frekvenciában eltérő komponense volt, a szűrő tudta detektálni mind a kettőt egy időben.

A fő nehézségeket az olyan jelek okozták, melyekben gyors, hirtelen hangváltások történtek, amelyek rövid ideig voltak állandóak, mert a szűrő nem tudott beállni a megfelelő alapfrekvenciára. Egy másik hátránya a szűrőnek, hogy a paramétereket a bejövő jelek előzetes ismerete szerint kell megválasztani, helyenként kísérletezni kell, hogy mik a legmegfelelőbb beállítások. A jövőben érdemes lehet úgy továbbfejleszteni az algoritmust, hogy automatikusan beállítsa a megfelelő paramétereket.

Köszönetnyilvánítás

Szeretnék köszönetet mondani mindenkinek, aki támogatott a dolgozat megírása során, családomnak, barátaimnak, kedvesemnek.

Külön köszönetet mondok konzulensemnek, dr. Rucz Péternek, aki mindvégig türelmes és segítőkész volt, nagyban segítette haladásomat.

Irodalomjegyzék

[1] O. Das, J. O. Smith és C. Chafe, „Improved Real-Time Monophonic Pitch Tracking,” 2020.

[2] O. Das, J. O. Smith és C. Chafe, „Real-Time Pitch Tracking in Audio Signals with the Extended Complex Kalman Filter,” 2017.

[3] H. B. Youngjoo Kim, „Introduction to Kalman Filter and Its Applications,” 8. május 2018. [Online]. Available: https://www.intechopen.com/books/introduction-and-implementations-of-the-kalman-filter/introduction-to-kalman-filter-and-its-applications.

[4] P. K. Dash, R. K. Jena, G. Panda és A. Routray, „An Extended Complex Kalman Filter for Frequency Measurement of Distorted Signals,” *IEEE Transactions on Instrumentation and Measurement,* pp. 1569-1574, 2000.

[5] A. d. Cheveigné és H. Kawahara, „YIN, a fundamental frequency estimator for speech and music,” 2001.